Int. J. Appl. Math. and Comp. Sci., 2000, Vol.10, No.4, 791–812

ABSTRACTION BASED CONNECTIONIST ANALOGY PROCESSOR

SYOZO YASUI*

The Abstraction Based Connectionist Analogy Processor (AB-CAP) is a trainable neural network for analogical learning/inference. An internal abstraction model, which extracts the underlying relational isomorphism and expresses predicate-argument bindings at the abstract level, is induced structurally as a result of the backpropagation training coupled with a structure-pruning mechanism. AB-CAP also develops dynamically abstraction and de-abstraction mappings for the role-filler matching. Thus, the propositions including both known and inferred ones can be expressed by, induced as, stored in and retrieved from the internal structural patterns. As such, there is no need for AB-CAP to use rule-based symbolic processing such as hypothesis making and constraint satisfaction or pattern completion checking. In this paper, AB-CAP is evaluated by using some examples. In particular, incremental analogical learning by AB-CAP shows that the internal abstraction model acquired from previous analogical learning acts as a potent attracter to bind a new set of isomorphic data, manifesting the analogical memory access/retrieval characteristics of AB-CAP.

Keywords: analogy, neural network, network pruning, dynamic binding, abstraction

1. Introduction

The analogy with water turns out to be helpful in understanding electrical current. The analogy is relational knowledge that allows one to find correspondences between aspects of two or more situations. The corresponding aspects are recognized as such when they play the same role in the relationships with other aspects within the respective situations (e.g., Charniak and Mcdermott, 1986). In the familiar example of Table 1, the two systems are structured in the same way, i.e., using a relational isomorphism. The analogical inference assumes that if two situations are known to be similar in some respects, it is likely that they will be similar in others (Luger and Stubblefield, 1993). For instance, if all facts except (nucleus, *attracts*, electron) have been taught and learned, then it would be an inference test to ask the question of (sun, *attracts*, ?). Analogies have been studied in various areas such as cognitive psychology, pedagogy, science history and epistemology. There also have been a

* Kyushu Institute of Technology, Faculty of Computer Science and Systems Engineering, Iizuka, 820 Japan, e-mail: yasui@ces.kyutech.ac.jp

great deal of computational research attempts. The structure mapping theory due to Gentner (1983) is particularly influential in this and other studies. The theory has been applied to develop SME (Structure Mapping Engine) by applying traditional AI techniques based on rule-based reasoning and symbolic processing (Falkenhainer *et al.*, 1989).

Learning by analogy in humans, on the other hand, often appears to occur spontaneously without logically thinking or analytical efforts. This aspect of analogy resembles sensory perception and seems to reflect something that characterizes brain mechanisms. Not surprisingly, therefore, the connectionist approach to the study of analogy has drawn considerable interest in recent years. ACME (Analogical Constraint Mapping Engine) due to Holyoak and co-workers is a well-known connectionist attempt (Holyoak and Thagard, 1989a; Holyoak *et al.*, 1994). Network nodes in ACME can represent potential correspondences of propositions across the source and target domains. The inter-node connections are made excitatory or inhibitory on the basis of structural, semantic and pragmatic criteria, and a relaxation algorithm is used to finalize the network dynamically. Although ACME is built on a neural network, its operational aspect is much AI-like, because of its frequent use of the if-then symbolic logic for comparison, constraint checking and hypothesis making. Thus, ACME operates essentially with logical induction, rather than learning in the usual connectionist sense, although related models such as IMM (Indirect Mapping Model) and LISA (Learning and Inference with Schemas and Analogies) adopt connectionism to a greater extent (Hummel *et al.*, 1994; Hummel and Holyoak, 1997). Besides the exposition of LISA, the last paper cited provides an intensive review on issues in analogy study from both cognitive science and connectionist standpoints. The Copycat model by Hofstadter and Mitchell (1994) should also be mentioned, although the kind of analogy it deals with is different from that of the present study, i.e., proportional analogies of the standard A:B::C:? form (e.g., doctor: patient :: teacher: ?), are considered in Copycat. The underlying idea of Copycat is more akin to the connectionist paradigm than to the symbolic one. Perhaps, the most interesting aspect of Copycat resides in its unique stochastic property that might have some parallel with mental fluidity of our brain.

Rumelhart (1989) expressed the need of a genuine connectionist approach whereby the goal, i.e., analogical reasoning or inference, should be achieved not as a result of explicitly intentional design but as a side effect of some general learning principle. In fact, Hinton (1986) described such a neural network to make inference on personal relationships in two isomorphic families, and the recognition of this early work in the context of analogy appears to be relatively recent (Holyoak *et al.*, 1994). With the same connectionist spirit but with a different architecture, this author presented an Abstraction Based Connectionist Analogy Processor (AB-CAP), which, like Hinton's network, assumes a relational isomorphism (Yasui, 1997b). As a result of the backpropagation training coupled with a mechanism to prune the AB-CAP structure dynamically, the underlying analogy structure (i.e., the internal abstraction model) is induced in AB-CAP more explicitly than in Hinton's network. Another notable aspect of AB-CAP in contrast to other connectionist attempts including Hinton's is that AB-CAP is ready to use for analogies involving multiple (three or more) analogs,

multiple analogies and incremental analogical learning (Yasui, 1998; Watanabe *et al.*, 1999). The knowledge data supplied to AB-CAP are not compared directly with each other, but the internal abstraction model induced in AB-CAP serves as a mediator in this context. Because of this property, AB-CAP circumvents combinatorial explosion which would be likely to arise in other existing analogy processors when applied to large-size problems.

The cognitive science community generally agrees that the analogy by humans involves semantic, pragmatic and structural faculties. AB-CAP, on the other hand, is purely structural; only the relational isomorphism is important, and semantic, pragmatic and other aspects are not considered. Obviously, therefore, AB-CAP has limitations as a model to understand human analogical learning. However, the internal abstraction model induced in AB-CAP is similar to the cognitive science notion of schema stemming from Kant. And schema induction in a computational framework has been a difficult problem of widely recognized importance, as reviewed by Hummel and Holyoak (1997). From this standpoint, AB-CAP is thought to be still relevant to analogy research in cognitive science, besides its potential technological applications, i.e., machine learning by analogy. In this paper, the performance of AB-CAP for analogical inference is examined using a family-tree example, more systematically than previously. Another example is given to examine incremental analogical learning by AB-CAP.

2. Basic Ideas and Archtecture of AB-CAP

The central idea underlying AB-CAP is obtained by noting that the solar/atom analogy (Rutherford analogy) of Table 1 can be generalized as shown in Fig. 1 and Table 2, wherein each pair of the corresponding specific items or objects (constants, e.g., sun and nucleus) are represented symbolically by a single abstracted item (variable, e.g. x) which plays a role (or roles) designated by a relational predicate (or predicates) such as *attract*, *revolve around* and so on. In Tables 1 and 2, the ordered triple (α, π, β) denotes a proposition that item α is related to item β through relational predicate π. The basic architecture of AB-CAP is shown in Fig. 2 and illustrated below using the solar/atom example. Throughout this paper, the terms 'unit', 'node' and 'neuron' are used synonymously. Also, 'slot' is often used to mean the input or output units. Every unit in the input and output layers is tagged by the name of a specific item (e.g., sun, earth, elrectron), and the input and output layers look the same.

Table 1. Solar/atom analogy: an example of a relational isomorphism.

Solar system:
(solar system, *greater than*, (sun, earth)) (sun, *included in*, solar system)
(sun, *greater than*, earth) (sun, *attract*, earth) (earth, *included in*, solar system)
(earth, *revolve around*, sun)
Atomic system:
(atom, *greater than*, (nucleus, electron)) (nucleus, *included in*, atom)
(nucleus, *greater than*, electron) (nucleus, *attract*, electron) (nucleus, *included in*, atom)
(electron, *revolve around*, nucleus)

Table 2. Generalization of the solar/atom analogy.

Abstraction model:
$(x,$ *greater than*, $(y,z))$ $(y,$ *included in*, $x)$ $(y,$ *greater than*, $z)$ $(y,$ *attract*, $z)$ $(z,$ *included in*, $x)$ $(z,$ *revolve around*, $y)$
Abstraction mapping:
solar system, atom $\rightarrow x$, sun, nucleus $\rightarrow y$, earth, electron $\rightarrow z$
De-Abstraction mapping:
$x \rightarrow$ solar system, atom, $y \rightarrow$ sun, nucleus, $z \rightarrow$ earth, electron

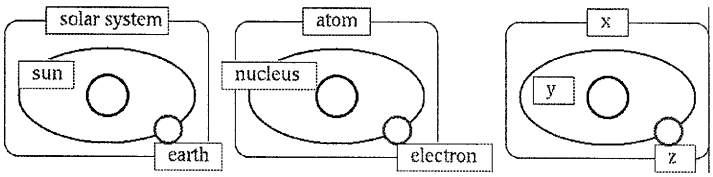

Fig. 1. Artist's concept of the solar/atom analogy and its generalization.

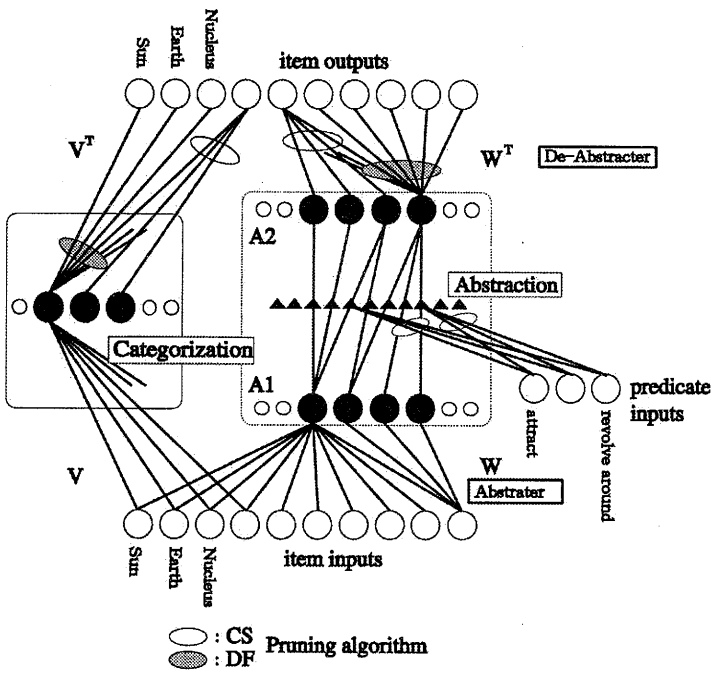

Fig. 2. Basic architecture of AB-CAP. The triangles represent the gating multipliers. CS (Convergence Suppression) and DF (Divergence Facilitation) fields (Yasui, 1997a) are present for the network pruning.

In AB-CAP, the specific items do not play a role by themselves. Instead, they let the corresponding abstracted items do the job. In this context, the specific items are indirect role fillers.

The possible relational predicates are treated as another set of inputs to AB-CAP. When a pair of specific item α and relational predicate π from the ordered triple (α, π, β) are provided as input, the two respective input neurons are set to '1' and the remaining input neurons '0' for their activities. AB-CAP is required to produce the activity of '1' in the output neuron corresponding to specific item β, and '0' in the rest of output slots. All neurons except for the item- and relation-input units have a sigmoidal activation function. All connections are allowed to have only non-negative weights. This is an appropriate restriction since what matters is whether there is a connection or no connection between any two nodes. All bias inputs are a negative constant (-4), except those to the final output units which are each independently updated by training and normally remain negative (with the initial value being -5).

The training is made using the backpropagation (BP) algorithm with β as the teacher signal for learning (α, π, β). Initially, a sufficiently large number of neurons are made available for use in each hidden layer. AB-CAP is pruned dynamically during the training by means of an algorithm called Convergence Suppression (CS) and Divergence Facilitation (DF) due to this author (Yasui, 1997a). Briefly, the CSDF algorithm is especially suitable for trimming multi-output neural networks such as AB-CAP. Like many other existing pruning methods, CSDF eliminates irrelevant and redundant connections or hidden units. CSDF does this in such a novel way as to minimize the total number of active hidden units and at the same time, to maximize the number of hidden units that can be used jointly by two or more output units. CS and DF are applied to certain receptive and projective fields as marked in Fig 2. No information is given to AB-CAP as to how many analogs (two in the solar-atom example) exist, as well as which analog individual items belong to. This class identification is handled by the auto-categorizer module that appears on the left in Fig. 2. Namely, the weight matrices $\boldsymbol{V}$ and $\boldsymbol{V}^T$ are expected to grow into the categorization and de-categorization mappings, respectively, and only the necessary and sufficient number of categorizer hidden units would survive the CSDF pruning.

The abstraction and de-abstraction mappings are formed through the development of the transpose pair of weight matrices $\boldsymbol{W}$ and $\boldsymbol{W}^T$, respectively. Thus, the AB-CAP architecture has some symmetry with respect to its upper and lower parts (Fig. 2). The structure that involves the two hidden layers, A1 and A2, is expected to grow into the internal abstraction model. The A1 and A2 hidden units that survive the CSDF pruning will correspond to the abstracted items such as x, y and z of Fig. 1 or Table 2. Each connection from the A1 to A2 layer is controlled by a weighted vector signal from the relational predicate inputs through gating multipliers. This predicate-argument binding is expressed by a three-dimensional matrix $\boldsymbol{U}$. As a result of all this, selective dynamic bindings will occur between abstracted items and relational predicate inputs. Accordingly, AB-CAP would generate its own version of the abstraction model such as the one given in Table 2.

3. Mathematical Description of AB-CAP

Further details of AB-CAP are given mathematically in this section.

3.1. Relational Isomorphism

Let $S_k = R \times A_k \times A_k$ $(k = 1, \ldots, N)$ be a situation. Here, $A_k = \{a_{k1}, \ldots, a_{kM}\}$ and $\cup_{k=1}^{N} A_k$ is the set of all distinct items. Furthermore, $R = \{r_1, \ldots, r_L, 0\}$ expresses all possible relational predicates including 'no relation' corresponding to the element 0. Thus, there are N categories corresponding to the N situations, each of which involves M distinct specific items. The situation S_k consists of all facts or propositions that belong to the k-th category. It is assumed in the present study that these individual propositions are expressible in the form $(\alpha,\ \pi,\ \beta)$ where $\alpha, \beta \in \cup_{k=1}^{N} A_k$. A relational isomorphism can be defined by introducing an abstract entity $B = \{b1, \ldots, b_M\}$. Thus, $S_1, \ldots, S_N$ are said to form a relational isomorphism iff

$$(\forall k\ \forall ((r_p, a_{ki}, a_{kj}) \in S_k)\ \exists m\ \exists n\,(r_p, b_m, b_n)) \wedge (\forall (r_p, b_m, b_n)$$

$$\exists i\ \exists j\ ((r_p, a_{1i}, a_{1j}) \in S_1) \wedge \cdots \wedge \exists i\ \exists j\ ((r_p, a_{Ni}, a_{Nj}) \in S_N)).$$

The situations $S_1, \ldots, S_N$ satisfying this isomorphism are said to be analogous to each other. If this is the case, each element of B is a variable which is a general symbolization of the corresponding specific items, and accordingly it is an abstracted item to be filled by the appropriate specific items. Thus, $R \times B \times B$ is a generalized situation, i.e., an abstraction model. In the solar/atom analogy, for example, we have $A_1 = \{\text{solar system, sun, earth}\}$, $A_2 = \{\text{atom, nucleus, electron}\}$, $R = \{\text{greater than, included in, attract, revolve around, 0}\}$, and $B = \{x, y, z\}$.

3.2. Abstraction Based Connectionist Analogy Processor (AB-CAP)

3.2.1. Structures Evolving into Abstraction

The activity vectors are denoted by $\boldsymbol{x} = [x_1, \ldots, x_{NM}]^T$ for the specific item neurons (e.g., sun, earth, nucleus, etc.) in the input layer and by $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_L]^T$ for the relational predicate units (e.g., *attract*, *revolve around*, etc.). The former signals are sent to two separate hidden layers, the categorizer hidden layer and the A1 layer. There are J A1 neurons ($J > M$) and the corresponding activity vector is denoted by $\boldsymbol{h} = [h_1, \ldots, h_J]^T \in [0, 1]^J$. These neurons are candidates of the nodes to represent the abstracted items. The transformation from $\boldsymbol{x}$ to $\boldsymbol{h}$ would evolve into the abstraction mapping. If $\boldsymbol{W} = [w_{ij}] \in [0, \infty]^{J \times NM}$ denotes the synaptic weight matrix associated with this transformation, then

$$h_i = f([\boldsymbol{W}\boldsymbol{x}]_i + c), \quad i = 1, \ldots, J.$$

Here, $[\boldsymbol{W}\boldsymbol{x}]_i$ stands for the i-th component of the vector $\boldsymbol{W}\boldsymbol{x}$, $f(\cdot) = \exp(\cdot)/(1 + \exp(\cdot))$ is the activation function, and c is a bias which is a negative constant.

Like A1 units, some units of the A2 layer are expected to survive the CSDF pruning and to represent the abstracted items. Signals from the A1 layer interact with the predicate inputs before arriving at the A2 layer (Fig. 2). This interaction facilitates predicate-argument bindings at the abstracted level and it is described, as illustrated below, by a three-dimensional weight matrix $\boldsymbol{U} = [u_{ijk}] \in [0, \infty]$, $i, j = 1, \ldots, J$, $k = 1, \ldots, L$. There are a total of L potentially useful connections from the i-th unit of the A1 layer to the j-th unit of the A2 layer. Each of these connections is gated by the relational predicate input vector $\boldsymbol{\pi}$ by using L multipliers corresponding to the L predicate inputs. Thus, the element u_{ijk} is the synaptic weight acting on the k-th predicate input before the multiplier. This arrangement, which is akin to the method based on the tensor product of vectors to represent the predicate-argument binding (Smolensky, 1990; Halford *et al.*, 1994), is expressed mathematically as follows. Let $\boldsymbol{u}_{ij} = [u_{ij1}, \ldots, u_{ijL}]$, so that a $J \times J$ matrix $\boldsymbol{P}[\boldsymbol{u}_{ij} \cdot \boldsymbol{\pi}]$ can be constructed where $\cdot$ denotes inner product of two vectors. If the vector $\boldsymbol{g} = [g_1, \ldots, g_J]^T \in [0, 1]^J$ denotes the activity pattern of the A2 hidden layer, then

$$g_i = f\big([\boldsymbol{Ph}]i + c\big), \quad i = 1, \ldots, J.$$

Note that

$$[\boldsymbol{Ph}]_i = \sum_{j=1}^{J} \sum_{k=1}^{L} u_{ijk} h_j \pi_k, \quad i = 1, \ldots, J.$$

Thus, each element of $\boldsymbol{U} = [u_{ijk}]$ quantifies the degree of binding involving a relational predicate and two arguments corresponding to two abstracted items, one in the A1 layer and the other in the A2 layer. Accordingly, the matrix $\boldsymbol{U}$ is what would evolve into the internal abstraction model after the BP-CSDF training which will be explained in Section 3.2.3.

3.2.2. Categorizer and De-Abstraction

Another hidden layer on the left in Fig. 2 constitutes the auto-categorizer to identify which of the classes (e.g., solar or atomic system) an input item belongs to. If $\boldsymbol{s} = [s_1, \ldots, s_K]^T \in [0, 1]^K$ is the activity vector of K categorizer hidden neurons ($K > N$), then

$$s_i = f\big([\boldsymbol{V}\boldsymbol{x}]i + c\big), \quad i = 1, \ldots, K,$$

where $\boldsymbol{V} = [v_{ij}] \in [0, \infty]^{K \times NM}$ is a weight matrix. The output layer provides the network's answer to (α, π, ?). Thus, it consists of $N \times M$ item neurons, like the item-input layer. If their activity vector is denoted by $\boldsymbol{y} = [y_1, \ldots, y_{NM}]^T \in [0, 1]^{NM}$, then

$$y_i = f\big([\boldsymbol{W}^T \boldsymbol{g}]_i + [\boldsymbol{V}^T \boldsymbol{s}]i + b\big), \quad i = 1, \ldots, NM.$$

Here, b is a bias. Unlike the bias c associated with other layers, b is not constant. The transformations from the A2 and categorizer hidden layers to the output layer are made by the matrices $\boldsymbol{W}^T$ and $\boldsymbol{V}^T$, respectively, in order for AB-CAP to generate the answer to (α, π, ?).

3.2.3. Backpropagation Training with Dynamic Structure Pruning

For each output unit, the error vector $\boldsymbol{e} = [y_1 - t_1, \ldots, y_{NM} - t_{NM}]^T$ is defined with $\boldsymbol{t} = [t_1, \ldots, t_{NM}]^T$ being the teacher signal vector. Let ω represent an element of $\boldsymbol{W}$, $\boldsymbol{V}$, $\boldsymbol{U}$ or b. Then ω is corrected iteratively by means of the backpropagation (BP) algorithm. Thus,

$$\Delta\omega_{\mathrm{new,BP}} = \eta\partial\boldsymbol{e}^T\boldsymbol{e}/\partial\omega + \alpha\Delta\omega_{\mathrm{old}},$$

where η and α are the learning-rate and momentum constants, respectively. $\partial\boldsymbol{e}^T\boldsymbol{e}/\partial\omega$ can be written down explicitly by using the delta rule as is well-known. Actually, the BP corrections for $\boldsymbol{W}$ and $\boldsymbol{V}$ are made twice in every iteration cycle, once for abstraction (i.e., as $\boldsymbol{W}$ and $\boldsymbol{V}$) and once for de-abstraction (i.e., as $\boldsymbol{W}^T$ and $\boldsymbol{V}^T$).

Furthermore, insignificant and irrelevant connections are eliminated by the CSDF pruning algorithm due to Yasui (1997a):

$$\Delta\omega_{\mathrm{new}} = \Delta\omega_{\mathrm{new,BP}} + \Delta\omega_{\mathrm{CS}} + \Delta\omega_{\mathrm{DF}},$$

with $\Delta\omega_{\mathrm{CS}}$ and $\Delta\omega_{\mathrm{DF}}$ being CS (Convergence Suppression) and DF (Divergence Facilitation) terms, respectively. $\Delta\omega_{\mathrm{new}}$ is ignored if it made the new value of ω become negative. There is no such restriction, however, for the variable bias b which normally remains negative. The CS mechanism is imposed across the receptive (fan-in) fields of all output neurons, and DF is introduced across the projective (fan-out) fields of the hidden-output connections (Fig. 2). Thus,

$$\Delta w_{ij,\mathrm{CS}} = -\varepsilon\,\mathrm{sgn}\,(w_{ij})\big(|w_{1j}| + \cdots + |w_{Jjz}| - |w_{ij}|\big)/J,$$

$$\Delta w_{ij,\mathrm{DF}} = -\gamma\partial\boldsymbol{e}^T\boldsymbol{e}/\partial w_{ij}\big(|w_{i1}| + \cdots + |w_{i,NM}| - |w_{ij}|\big)/(NM),$$

for $j = 1, \ldots, NM$ and $i = 1, \ldots, J$. Here, ε and γ are algorithm parameters. The expressions are similar for $\Delta v_{ij,\mathrm{CS}}$ and $\Delta v_{ij,\mathrm{DF}}$ associated with the categorizer module. In addition, the CS pruning is applied also to u_{ijk}'s, which are the gating weight parameters to allow for interactions between the abstracted items and predicate inputs, such that

$$\Delta u_{ijk,\mathrm{CS}} = -\varepsilon\,\mathrm{sgn}\,(u_{ijk})\big(u_{ij1} + \cdots + u_{ijL} - u_{ijk}\big)/L, \quad i,j = 1, \ldots, J.$$

3.3. Parameter Setting/Control and Initial Conditions

Generally speaking, AB-CAP is considerably robust against the choice of its parameter values. It should be noted, however, that the shorter the learning time is, the more often the learning fails. The present parameter setting is a compromise between the learning speed and accuracy. The learning-rate constant (η) was 5 for $\boldsymbol{W}^T$ and $\boldsymbol{V}^T$, and 5 for $\boldsymbol{W}$ and $\boldsymbol{V}$. Moreover, η for $\boldsymbol{U}$ was 10 and 20 for Phase 1 and 2, respectively. Here, Phase 1 refers to the training period before all output errors become less than 0.95 in the absolute value, and the remaining period is denoted by Phase 2. Normally, the desirable internal structure is basically formed near the end of Phase 1,

and Phase 2 accelerates and shapes up the emergent structure. The momentum constant (α) was 0.2. η for the bias weight associated with each output unit was 0 and 1 for Phase 1 and 2, respectively. As regards the CSDF pruning, the facilitation parameter (γ) was 5, and the suppression parameter (ε) for both $\boldsymbol{W}$ and $\boldsymbol{V}$ was 0.002 for Phase 1 and 0.005 for Phase 2. ε for $\boldsymbol{U}$ was 1×10^{-6} and 5×10^{-3} for Phase 1 and 2, respectively. For Phase 2, however, the convergence suppression (CS) included 'self-suppression'. That is, the last equation was slightly modified such that

$$\Delta u_{ijk,\mathrm{CS}} = -\varepsilon \operatorname{sgn}(u_{ijk})(u_{ij1} + \cdots + u_{ijL})/L, \quad i,j = 1,\ldots,J.$$

This modification has been found to be more effective for scavenging remnants in the u_{ijk} values which would otherwise remain unvarnished even near the end of the BP-CSDF training. The reason for this effectiveness is as follows. If one and only one unnecessary connection happens to survive the CSDF pruning in a fan-in field, then no inhibitory force would be available from the 'non-self-suppression scheme' to eliminate that connection. As for the non-negative weight parameters, the initial values of $w_{ij} \in \boldsymbol{W}$ were 0 and those of $v_{ij} \in \boldsymbol{V}$ and $u_{ijk} \in \boldsymbol{U}$ were distributed randomly in $[0,1]$, $\forall i,j,k$. In order to prevent the weight values from going too wild during the training, a restriction was imposed such that $w_{ij} < 6$, $v_{ij} < 6$, $u_{ijk} < 10$ for Phase 1, and $w_{ij} < 12$, $v_{ij} < 10$, $u_{ijk} < 20$ for Phase 2.

4. Analogical Inference by AB-CAP

If some relationships are excluded from the training dataset, then the inference tests asking (α, π, ?) can be made about those untaught relations after the training. A success of the inference by AB-CAP entails the proper acquisition of the abstraction ($\boldsymbol{W}^T$) and de-abstraction ($\boldsymbol{W}$) mappings, and the categorization ($\boldsymbol{V}^T$) and de-categorization ($\boldsymbol{V}$) mappings, as well as the abstraction model determined by the predicate-argument binding ($\boldsymbol{U}$) at the abstracted level. The performance of AB-CAP for analogical inference was demonstrated by the previous studies using the solar/atom (Yausi, 1998), family-tree (Yasui, 1997b; 1998), and geological (Yausi, 1998) examples. The purpose of this section is a closer evaluation by revisiting the family-tree example of Fig. 3 or Table 3.

4.1. Inference Test Protocol

In the family-tree example of Fig. 3 or Table 3, five family roles are considered. These are *husband*, *wife*, *father*, *mother* and *child*, so that there are a total of thirty six personal relationships, eighteen for each family. The number of hidden units initially made available for use was eight for each of the A1 and A2 hidden layers and also eight for the categorizer layer. The inference tests were made when the CSDF-BP learning was complete, i.e., when all output errors were < 0.2 in the absolute value. The inference was judged as successful when the output error for every untaught proposition was < 0.3 in the absolute value. The inference failure includes the training runs that never learned the training data before 80,000 iterations. Most (85%) successful runs ended after 35,000–50,000 iterations. Note that, in a previous

paper (Yasui, 1998), a table similar to Table 3 suffers from a careless error of writing, for instance, (John, *wife*, Nancy), instead of (John, *husband*, Nancy).

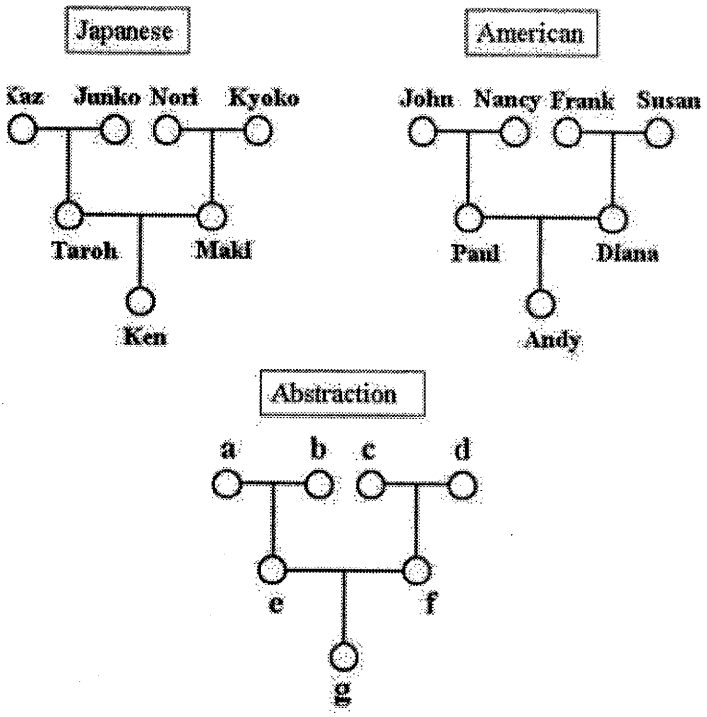

Fig. 3. Two relational isomorphic families and their abstraction.

Table 3. A written version of Fig. 3.

Japanese family relations:
(Kaz, *husband*, Junko) (Junko, *wife*, Kaz) (Junko, *mother*, Taroh) (Taroh, *child*, Kaz) (Maki, *wife*, Taroh) (Ken, *child*, Maki), etc.
American family relations:
(John, *husband*, Nancy) (Nancy, *wife*, John) (Nancy, *mother*, Paul) (Paul, *child*, John) (Diana, *wife*, Paul) (Andy, *child*, Diana), etc.
Abstraction model:
(a, *husband*, b) (b, *wife*, a) (b, *mother*, e) (e, *child*, a) (f, *wife*, e) (g, *child*, f), etc.
Abstraction mapping:
Kaz, John $\rightarrow$ a, Junko, Nancy $\rightarrow$ b, ..., Ken, Andy $\rightarrow$ g
De-Abstraction mapping:
a $\rightarrow$ Kazuo, John, b $\rightarrow$ Junko, Nancy, ..., g $\rightarrow$ Ken, Andy

4.2. Inference Performance

The inference test results are shown in Fig. 4. For comparison, the inference test was also performed without the CSDF pruning, i.e., BP only. The inference success rate was examined as a function of the number of relational propositions excluded from the training dataset. The removal was made from one family only, or alternatively, evenly from both families. The two alternatives, however, produced no significant difference in the result. Actually, the untaught relations (i.e., the test dataset) did not include a counterpart pair such as (Kaz, *husband*, Junko) and (John, *husband*, Nancy). If these data were both removed from the training dataset, then AB-CAP would be guaranteed to fail in answering to either (Kaz, *husband*, ?) or (John, *husband*, ?). Also, the selection for the removal from the training dataset was controlled so as to avoid the cases where a person would never appear as a teacher signal. Although the training needed a longer time and the inference success rate decreased somewhat, AB-CAP managed to handle such cases so long as the person appeared in an input slot. In a separate experiment, 'no answer propositions' such as (John, mother, 'none') were included in the training dataset, so that the teacher signal was zero for all the output units. The inference success rate was improved somewhat unless the percentage of no-answer propositions mixed in the training dataset was not too high. Thus, there appears to be some optimum increase in such negative information for the training.

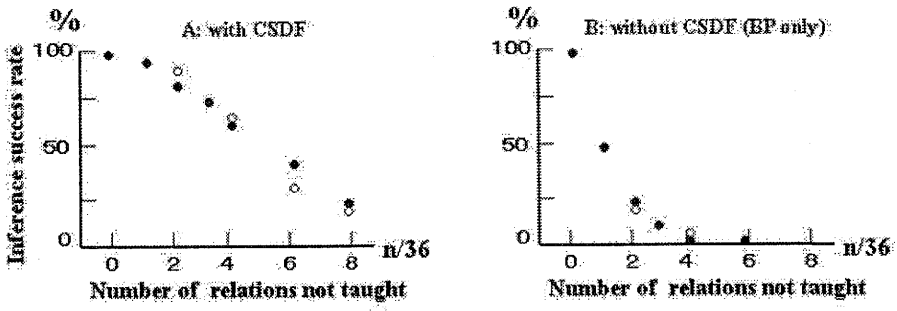

Fig. 4. Inference success rate vs. the number (n) of relational propositions removed from the training dataset for 100 runs for each data point. Filled circles: removal exclusively from one family. Open circles: removal evenly from both families ($n/2$).

4.3. Evolution of the Internal Structure

Figure 5 shows how AB-CAP evolved in a successful case of the family-tree learning. In this record, 'max_error' refers to the largest of the absolute output errors in the last 1,000 iterations. The 3D profiles of the predicate-argument binding matrix $\boldsymbol{U} = [u_{ijk}]$ are presented in the following way.

The five possible predicates ($k = 1, \ldots, 5$) relating two abstracted items i and j are coded in an alphabet, i.e., *father* $\rightarrow$ F, f, *mother* $\rightarrow$ M, m, *husband* $\rightarrow$ H, h, *wife* $\rightarrow$ W, w, *child* $\rightarrow$ C, c. The capital letter is displayed if the corresponding u_{ijk}

value exceeds the higher threshold, th_1. The lower-case letter is used if the value falls between th_1 and the lower threshold, th_2. No letter appears in the slot if $u_{ijk} < th\_2$. Starting from the initial random state (see A), a structure began to appear in $\boldsymbol{V}$ such that two of the eight available categorizer hidden units became selectively active (corresponding to the Japanese and American) and the CSDF pruning eliminated the rest (see B). At C, the matrix $\boldsymbol{W}$, the abstraction and de-abstraction mappings was formed almost properly, but the gated binding represented by $\boldsymbol{U}$ was not yet established. At D, max_error dropped nearly to the zero level, and the learning successfully ended. Thus, seven units in each of the A1 and A2 layers finally survived the pruning to represent the seven abstracted persons, a, ..., g of Fig. 3 or Table 3. In this example, a total of eight units were made available in each of the two layers. This is only one unit more than necessary. This saving was made in order to make the $\boldsymbol{U}$ result compact in size, simply for the sake of illustration such as Fig. 5. The final array $\boldsymbol{W}$ consists of two identical patterns. One reflects the abstraction mapping from the Americans and the other from the Japanese, each to the corresponding abstracted persons (see Table 3). The panel (D) of Fig. 5 is enlarged as given in Figure 6, showing that the internal abstraction model (i.e., $\boldsymbol{U}$ matrix) finally captured conforms to the abstraction described in Table 3, i.e., (a, *husband*, b), (b, *wife*, a), (e, *mother*, g) and so on.

4.4. Discussion

As is evident in the present example, the CSDF pruning plays a crucial role in AB-CAP to form the final internal structure properly. Figure 7 shows a diagram that can be drawn from the final structure of Fig. 6, illustrating schematically how AB-CAP can successfully produce the right output of 'John' in response to (Nancy, *wife*, ?), i.e., whose wife is Nancy? In this example, the item input is signaled from the 'Nancy' neuron to the 'b' neuron, and only this unit gets activated in the A1 layer since its input signal is amplified by a large weight to overcome the threshold set by the bias input to the 'b' unit. The relational predicate signal from the *wife* slot goes to the gating multipliers associated with all the potentially relevant abstracted propositions, i.e., (b, *wife*, a), (d, *wife*, c), and (f, *wife*, g) (see Fig. 3). However, no signal comes from either 'd' or 'f' neuron of the A1 layer. In the A2 layer, therefore, the 'a' neuron is the only unit that receives a supra-threshold signal to get activated as illustrated in Fig. 7. This activation signal goes to the two output slots, one tagged as John and the other Kaz. But only the 'John' neuron can be activated, since it receives the signal from 'American' neuron whereas the input to the 'Kaz' neuron is not large enough to overcome the threshold, due to no signal from the categorization layer.

Actually, the data of (Nancy, *wife*, John) were not included in the training dataset for the experiment of Fig. 5. Nevertheless, this whole proposition was induced structurally in AB-CAP (Figs. 6 and 7). Thus, AB-CAP after the training knew that Nancy is John's wife, which is knowledge never told to AB-CAP. This inference is a knowledge transfer from the Japanese counterpart, i.e., (Junko, *wife*, Kaz). As is evident from Figs. 6 and 7, however, this transfer is indirect since it is relayed only through the abstracted proposition (b, *wife*, a). That is, a micro-structure expressing

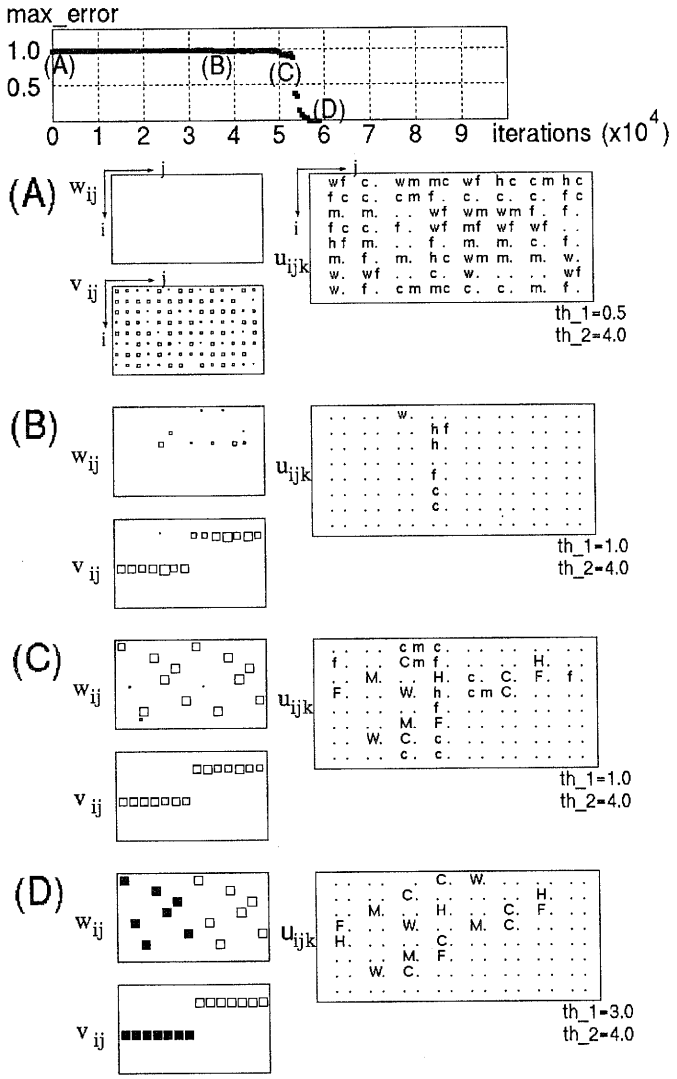

Fig. 5. Evolution of the internal structure during a successful training run for the family-tree example of Fig. 3. The size of each square is proportional to the value of the corresponding element in the abstraction mapping ($\boldsymbol{W}$) or categorization mapping ($\boldsymbol{V}$) matrix. The 3D matrix ($\boldsymbol{U}$) describing the internal abstraction model is expressed symbolically in an alphabet, e.g., *father* $\rightarrow$ F, f (see the text).

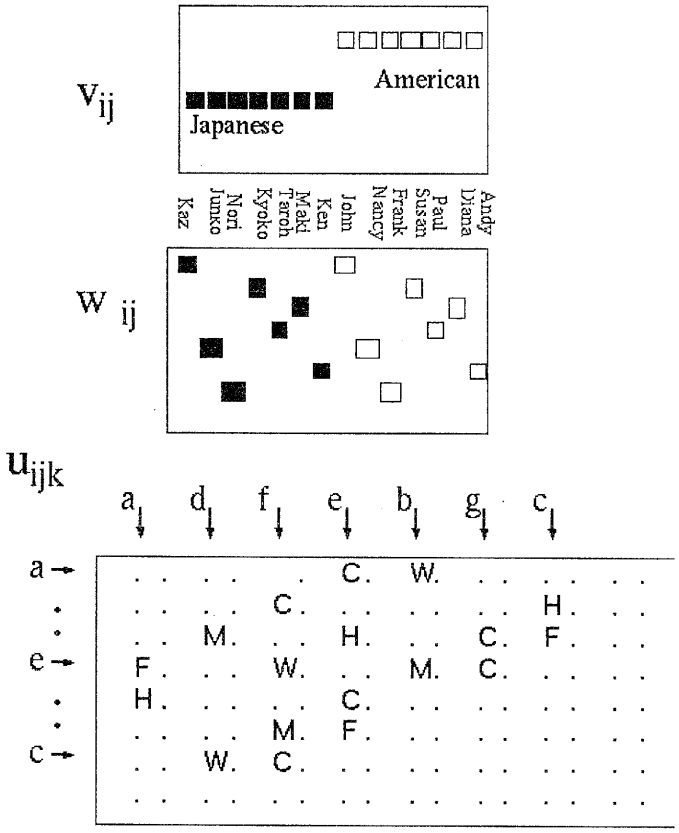

Fig. 6. Enlargement of Fig. 5(D) illustrating the post-training structure of AB-CAP learning in the family-tree example of Fig. 3. In this particular run, the seven abstracted persons in the $\boldsymbol{U}$, b, d, f, e, b, g, c corresponded to $i, j = 1, 2, \ldots, 7$, in that order.

(b, *wife*, a) was induced by (Junko, *wife*, Kaz) in the absence of (Nancy, *wife*, John). While this process was going, the abstraction binding as well as de-abstraction binding between the 'b' and 'Nancy' units were induced by those training data that involved Nancy, i.e., (John, *husband*, Nancy), (Paul, *child*, Nancy) and (Nancy, *mother*, Paul). The binding between the 'a' and 'John' units was made in a similar fashion.

Since the whole proposition (Nancy, *wife*, John) was induced structurally in AB-CAP, one can just look at the post-training structure (Fig. 6) to answer not only to (Nancy, *wife*, ?), but also to (?, *wife*, John), i.e., who is John's wife? or (Nancy, ?, John), i.e., what family role does Nancy play for John? In fact, such visual inspection has been recently computerized (Fujimura and Yasui, 2000). It is also obvious by looking at the structure that John is the answer to the inquiry asking who in the American family plays the same role as Kaz in the Japanese family. Again, it would

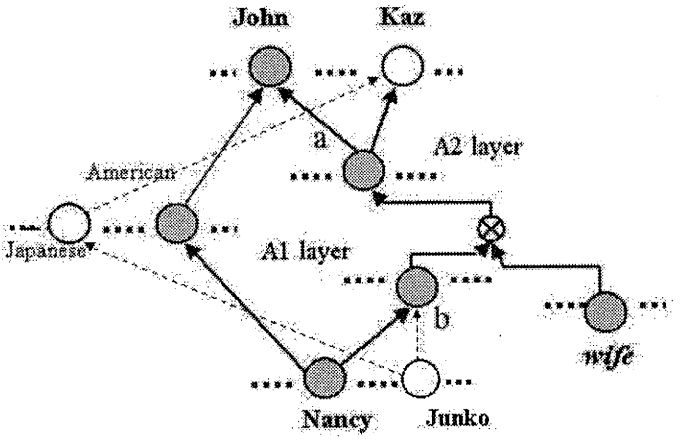

Fig. 7. Schematic diagram obtained from Fig. 6, illustrating how the optimum (necessary and sufficient) structure operates in AB-CAP to give the right answer 'John' to the question (Nancy, *husband*, ?). Dark circles are the units activated by the question. The encircled cross symbol represents the gating multiplier for the predicate-argument binding in the internal abstraction model. Although not shown here, there is a negative-valued bias input to each neuron except for the input units.

be a straightforward work to write out a computer program that can do this by reading the post-training structure.

It should also be noted that a successful inference was almost always (94%) accompanied by the post-training internal structure (e.g., Fig. 6) that captured the abstraction scheme of Table 3.

5. Incremental Analogical Learning by AB-CAP

The purpose of this section is to examine what would happen if AB-CAP received a new set of data having the same relational isomorphism as those previously learned. This is the problem of incremental analogical learning which was examined previously in AB-CAP by using an analogy concerning geological orientation (Yasui, 1998). Here, a different example documented in Table 4 is used.

5.1. Performance Evaluation

A preliminary test was made first for simultaneous (non-incremental) learning of all the three analogs, i.e., water, electric and heat systems. The inference test score was good (70–95% success) as far as the number of relationships removed from the training dataset was no more than four. Figure 8 shows a typical record showing the progress of incremental analogical learning by AB-CAP. The training was made first by teaching two of the three analogs. They were the water and electric systems in the experiment of Fig. 8. When the learning was completed successfully around stage C,

Table 4. Water/electric/heat analogy.

Water Flow:	
(spring, *higher than* (in altitude), lake)	(spring, *supply*, water)
(river, *conduct*, water)	(river, *follow*, spring)
(lake, *follow*, river)	(lake, *accumulate*, water)
(water, *flow through*, river)	(water, *flow to*, lake)
Electric Current:	
(battery, *higher than* (in voltage), capacitor)	(battery, *supply*, charge)
(register, *conduct*, charge)	(resister, *follow*, battery)
(capacitor, *follow*, resister)	(capacitor, *accumulate*, charge)
(charge, *flow through*, resister)	(charge, *flow to*, capacitor)
Heat Flow:	
(fire, *higher than* (in temperature), coffee)	(fire, *supply*, heat)
(metal, *conduct*, heat)	(metal, *follow*, fire)
(coffee, *follow*, metal)	(coffee, *accumulate*, heat)
(heat, *flow through*, metal)	(heat, *flow to*, coffee)
Abstraction Model:	
(a, *higher than*, c)	(a, *supply*, d)
(b, *conduct*, d)	(b, *follow*, a)
(c, *follow*, b)	(c, *accumulate*, d)
(d, *flow through*, b)	(d, *flow to*, c)
Abstraction Mapping:	
spring, battery, fire → a,	river, resister, metal → b,
lake, capacitor, coffee → c,	water, charge, heat → d
De-Abstraction Mapping:	
a → spring, battery, fire,	b → river, resister, metal,
c → lake, capacitor, coffee,	d → water, charge, heat

the third analog which is the heat system was added to the training dataset. The output error rose immediately, but quickly returned to the zero level at D and the incremental analogical learning was successful.

5.2. Development of the Internal Structure

Next, the record of Fig. 8 is examined for evolution of the internal structure. The same convention as in Fig. 5 is used here to present the predicate-argument binding matrix $\boldsymbol{U}$, with the alphabet coding being such that *accumulate* → A, a, *conduct* → C, c, *higher than* → H, h, *supply* → S, s, *follow* → F, f, *flow through* → W, w, *flow to* → O, o.

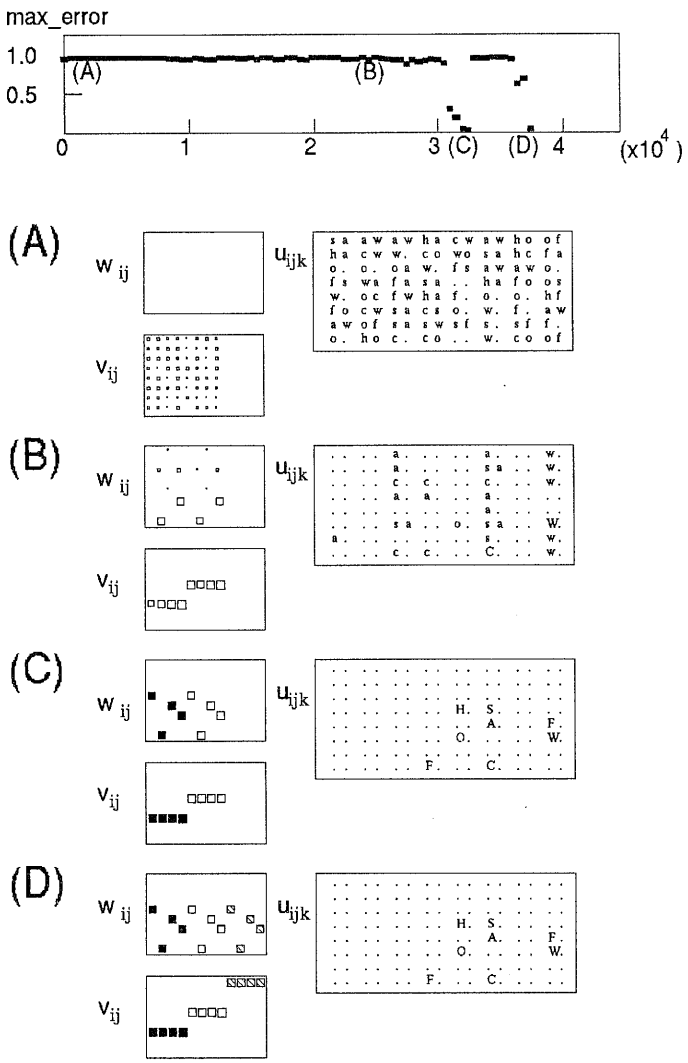

Fig. 8. Evolution of the AB-CAP internal structure during incremental analogical learning of the heat system after analogical learning of the water and electric systems.

The record of $\boldsymbol{V}$ at stage C shows that two categorizer units survived the CSDF pruning, representing the water and electric systems. Correspondingly, the record of $\boldsymbol{W}$ at C shows two identical connection patterns. The record of $\boldsymbol{U}$ at C expresses the internal abstraction model which was induced from learning the water and electric systems. The two $\boldsymbol{W}$ patterns at C manifest correspondence bindings between specific and abstracted items. At D, the $\boldsymbol{V}$ record shows the appearance of one more

categorizer hidden unit, reflecting the addition of the heat system to the training dataset. Furthermore, at D, the panel of $\boldsymbol{W}$ shows the emergence of a new pattern. But this one is identical to the two existing patterns formed at C. The whole pattern of $\boldsymbol{U}$, however, remains unchanged since its formation at C. All this implies that the internal abstraction model acquired from the prior analogical learning remains undisturbed and acts as an attracter to capture new isomorphic knowledge such as the heat system of the present example.

6. Closing Remarks

AB-CAP (Abstraction Based Connectionist Analogy Processor) is one way to computerize abstraction schemes such as the one presented in Table 3. Its internal abstraction model has some parallel with the notion of *schema* in cognitive science. It can also be said that AB-CAP is a product of the 'structurism' stemming from Gentner (1983). The semantic and pragmatic aspects and background knowledge as well as surface similarities are not considered in AB-CAP, whereas human analogical reasoning is known to use these cues often. In AB-CAP, only a relational isomorphism is what is needed. Thus, the items and relational predicates *per se* may have no semantic meaning in AB-CAP, so that they can be just symbols or numbers as in an exercise presented earlier (Yasui, 1997b). Thus, it is beyond the capability of AB-CAP to induce, for instance, the notion of 'marriage' from the training dataset used in the family-tree example. For a similar reason, it does not matter to AB-CAP whether the analogy is of a within-domain or a between-domain type (cf., e.g., Vosniadou and Ortony, 1989). As such, AB-CAP has limitations as a model for analogical learning by humans, although AB-CAP might lead to technological applications, i.e., machine learning by analogy. Nevertheless, AB-CAP may still have relevance to important issues in cognitive science, since it is based on non-intentional mechanisms to form, maintain and retrieve the analogical memory. The key to this capability consists of the internal abstraction model and abstraction/de-abstraction mappings. These are all induced automatically in AB-CAP as a result of the BP-CSDF training. The CSDF pruning mechanism plays a crucial role in this process. This agrees with the connectionist engineer's notion often referred to as PRR (Principle of Redundancy Reduction) that good generalization ability of a neural network entails its compact internal representation.

6.1. No Direct Processing between Source and Target in AB-CAP

In the study of analogy, the two terms, 'source' and 'target', have traditionally been used, like a analogical transfer or pattern matching between a source and a target. The source-target notion is somewhat obscured in AB-CAP as well as other contemporary models, since the data can be fragmentary in both the 'source and target'. More importantly, in AB-CAP there is no direct interaction or comparison between the 'source and target'. The abstraction model that emerges in AB-CAP acts as a mediator in this context. An analogy processor called IMM due to Hummel *et al.* (1994) mentioned in the Introduction uses a form of guided retrieval to indirectly establish

mappings between the source and target elements. In AB-CAP, on the other hand, source-target mappings in themselves do not exist. In AB-CAP, predicate-argument bindings are made on abstracted items instead of specific items, so as to generate abstracted propositions such as (z, *revolve around*, y) in Table 2. For this reason, the risk of a combinatorial explosion is greatly reduced for large-size application problems. In fact, AB-CAP allows for multiple analogs and/or multiple analogies to be processed at the same time without significant computational pressure (Yasui, 1998; Watanabe *et al.*, 1999). In AB-CAP with these characteristic features, the constraint satisfaction and isomorphic pattern completion are automatically accomplished without explicit manipulations such as hypothesis making/checking.

6.2. Propositions are Embedded as Micro-Structures in AB-CAP

In this and earlier studies, the analogical inference by AB-CAP has been examined by removing a proposition, say, (Nancy, *wife*, John) from the training dataset. AB-CAP after the training generates the answer 'John' to the question (Nancy, *wife*, ?). Actually, by looking at the post-training structure, one can at once answer correctly not only to (Nancy, *wife*, ?) but also to (?, *wife*, John) or (Nancy, ?, John). This is because what is induced in AB-CAP is a micro-structure expressing the entire proposition (Nancy, *wife*, John). This point is basically valid for Hinton's connectionist model (1986), and the following remark quoted from Holyoak *et al.* (1994) appears somewhat misleading; "...Hinton's generalization task involved giving the system the first argument and relation and asking it to generate the second argument, whereas ACME was asked to generate entire new propositions...". Also by visual inspection on the post-training micro-structures of AB-CAP, one can easily tell, for instance, who in the Japanese family plays the same role as John. Such visual inspection can be computerized easily. Thus, both known and inferred correspondences, no matter whether they are concerned with single items or relations of items, are embedded in the structure. In ACME and some other models, the propositions are represented by single nodes. In AB-CAP, on the other hand, propositions are expressed by, induced as, stored in, and retrieved from structural patterns.

6.3. Dynamic Binding and Access/Retrieval Mechanisms in AB-CAP

The performance quality of AB-CAP regarding analogical memory access and retrieval can be demonstrated particularly well by incremental analogical learning. It is shown in this and earlier studies (Yasui, 1998; Watanabe *et al.*, 1999) that the internal abstraction model obtained from prior analogical learning is a potent attracter that binds dynamically new datasets of the same isomorphic structure. In recent years, *dynamic binding* has drawn increasing recognition as a fundamental mechanism for cognitive functions of the brain. And Hummel *et al.* (1994) appear to be the first to introduce this notion to analogy research, and predicate-argument dynamic binding is an operational strategy in LISA, a model for human analogical learning (Hummel and Holyoak, 1997). Dynamic bindings in AB-CAP work at somewhat different levels and in a different context in comparison to LISA. In AB-CAP, dynamic bindings occur between the relational predicates (roles) and the abstracted items (role fillers), as

a process to acquire the internal abstraction model, i.e., the 3D predicate-argument binding matrix $\boldsymbol{U}$. Also, dynamic bindings in AB-CAP are coupled with the CSDF pruning to develop the correspondence matching between the specific and abstracted items, as is expressed by the abstraction ($\boldsymbol{W}$) or de-abstraction ($\boldsymbol{W}^T$) mapping. In a sense, therefore, the specific items in AB-CAP do not play a role by themselves. Instead, they let their representatives (i.e., abstracted items) be the actual role fillers.

In a case study of incremental analogy described elsewhere (Watanabe *et al.*, 1999), AB-CAP was trained to learn in succession two analogies of the solar/atom and water/electric systems, so that two internal abstraction models were induced. After that, a training dataset from the heat system was added. As a result, new bindings were formed to ($\boldsymbol{W}$) and from ($\boldsymbol{W}^T$) the existing abstraction model corresponding to the water/electric analogy, not the other one generalizing the solar/atom analogy. This is selective retrieval of analogical memory or allocation of new role-fillers, and is reminiscent of human analogical learning whereby one often recognizes a new situation as analogous to one of the relational concepts previously learned. From such a viewpoint, the internal abstraction models would be stored as a part of the long-term memory and the abstraction and de-abstraction mappings would pertain to the working memory.

6.4. Rutherford and AB-CAP

In reality, the situations faced by the learner are unlikely to conform to the ideal isomorphism assumed in this study. The isomorphism is probably obscured by irrelevant information, and potential analogs would need to be recognized before discovery of the analogy. This noise-filtering problem appears to remain relatively unexplored, despite its obvious importance in analogy research. The problem has recently been examined in the computational framework of AB-CAP. The result obtained by using the Rutherford analogy shows that the performance of AB-CAP is not affected by the presence of irrelevant objects such as the moon, positron and neutron (Yasui and Watanabe, 2000).

In the present study, AB-CAP learns the solar and atomic systems simultaneously. This was not exactly the case of Rutherford. At least initially in his and others' mind at that time, the solar system was just a knowledge having nothing to do with analogy. Only retrospectively can it be called a prior example or source analog. Perhaps in Rutherford's brain, knowledge of the solar system was retrieved from the long-term memory to provide input data to his abstraction machinery when the physicist was confronted by fragmentary information about the atom. In this context, Rutherford might have been like AB-CAP capturing the abstraction model by learning the solar and atomic systems simultaneously.

In reality, an isomorphism may also be incomplete or inaccurate. The celebrated tumor/fortress analogy which is one such example can be understood by most people (e.g., Holyoak and Thagard, 1989b). Again, AB-CAP remains to be investigated regarding this kind of non-ideal relational isomorphism. Future study may also include a possible extension of AB-CAP for dealing with high-order propositions such as (Tom, *know*, (John, *love*, Nancy)) or ((Car A, *hit*, Car B), *cause*, (Car B, *hit*, Car C)).

References

Charniak E. and Mcdermott D. (1986): *Introduction to Artificial Intelligence.* — Reading, MA: Addison-Wesley.

Gentner D. (1983): *Structure mapping: A theoretical framework.* — Cognit. Sci., Vol.7, pp.155–170.

Falkenhainer B., Forbus K.D. and Gentner D.(1989): *The structure-mapping engine.* — Artif. Intell., Vol.41, No.1, pp.1–63.

Fujimura H. and Yasui S. (2000): *Connectionist analogical inference on predicates and their arguments.* — Proc. *ICONIP 2000*, Taejon, Korea, pp.482–487.

Halford G.S., Wilson W.H., Guo J., Gayler R.W., Wiles J. and Stewart J.E.M. (1994): *Connectionist Implications for Processing Capacity Limitations in Analogies*, In: Advances in Connectionist and Neural Computation Theory, Vol.2, Analogical Connections (K.J. Holyoak and J.A. Barnden, Eds.). — Norwood, NJ: Ablex Publishing Co., pp.363–415.

Hinton G.E. (1986): *Learning distributed representations of concepts.* — Proc. 8th Ann. Conf. *Cognitive Science*, Hillside, NJ, pp.1–12.

Holyoak K.J. and Thagard P. (1989a): *Analogical mapping by constraint satisfaction.* — Cognit. Sci., Vol.13, pp.295–355.

Holyoak K.J. and Thagard P. (1989b): *A computational mode of analogical problem solving*, In: Similarity and Analogical Reasoning (S. Vosniadou and A. Ortony, Eds.). — Cambridge: Cambridge Univ. Press, pp.241–266.

Holyoak K.J., Novick L.R. and Melz E.R. (1994): *Component processes in analogical transfer: Mapping, pattern completion and adaptation*, In: Advances in Connectionist and Neural Computation Theory (K.J. Holyoak and J.A. Barnden, Eds.), Vol.2, Analogical Connections. — Norwood, NJ: Ablex Publishing Co., pp.113–180.

Hummel J.E., Burns B. and Holyoak K.J. (1994): *Analogical mapping by dynamic binding: Preliminary investigations*, In: Advances in Connectionist and Neural Computation Theory (K.J. Holyoak and J.A. Barnden, Eds.), Vol.2, Analogical Connections. — Norwood, NJ: Ablex Publishing Co., pp.416–445.

Hummel J.E. and Holyoak K.J. (1997): *Distributed representations of structure: A theory of analogical access and mapping.* — Psych. Rev., Vol.104, No.3, pp.427–466.

Luger G.F. and Stubblefield W.A. (1993): *Artificial Intelligence.* — Redwood City, CA: Addison-Wesley.

Rumelhart D.E. (1989): *Toward a microstructural account of human reasoning*, In: Similarity and Analogical Reasoning (S. Vosniadou and A. Ortony, Eds.). — Cambridge: Cambridge Univ. Press, pp.298–312.

Smolensky P. (1990): *Tensor product variable binding and the representation of symbolic structures in connectionist systems.* — Artif. Intell., Vol.46, pp.159–216.

Vosniadou S. and Ortony A. (1989): *Similarity and analogical reasoning: A synthesis*, In: Similarity and Analogical Reasoning (S. Vosniadou and A. Ortony, Eds.). — Cambridge: Cambridge Univ. Press, pp.1–17.

Watanabe T., Fujimura H. and Yasui S. (1999): *Connectionist incremental learning by analogy.* — Proc. *ICONIP'99* Conf., Perth, Australia, Vol.III, pp.940–945.

Yasui S. (1997a): *Convergence suppression and divergence facilitation: Minimum and joint use of hidden units by multiple outputs.* — Neural Networks, Vol.10, No.2, pp.353–367.

Yasui S. (1997b): *Connectionist analogical inference in relational isomorphism paradigm.* — Proc. *ICONIP'97* Conf., Dunedin, New Zeland, Vol.1, pp.526–530.

Yausi S. (1998): *Connectionist abstraction for machine learning by analogy.* — Proc. *ICONIP'98* Conf., Ohmsha, Japan, Vol.3, pp.1453–1458.

Yasui S. and Watanabe T. (2000): *Connectionist analogical learning of relational isomorphism obscured by irrelevant relationships.* — Proc. 6th Int. Conf. *Soft Computing*, Iizuka 2000, Japan (CD-ROM).